Computational Biology and Chemistry 57 (2015) 80–92

Contents lists available at ScienceDirect

Computational Biology and Chemistry

journal homepage: www.elsevier.com/locate/compbiolchem

Research Article

Ancestral population genomics using coalescence hidden Markov models and heuristic optimisation algorithms

Jade Yu Cheng*, Thomas Mailund

Bioinformatics Research Centre, Aarhus University, C.F. Møllers Allé 8, 8000 Aarhus, Denmark

ARTICLE INFO

Article history:
Received 7 January 2015
Accepted 2 February 2015
Available online 5 March 2015

Keywords:
Sequential Markov coalescence
Coalescent hidden Markov models
Demographic inference
Numerical optimisation
Genetic algorithm
Particle swarm optimisation

ABSTRACT

With full genome data from several closely related species now readily available, we have the ultimate data for demographic inference. Exploiting these full genomes, however, requires models that can explicitly model recombination along alignments of full chromosomal length. Over the last decade a class of models, based on the sequential Markov coalescence model combined with hidden Markov models, has been developed and used to make inference in simple demographic scenarios. To move forward to more complex demographic modelling we need better and more automated ways of specifying these models and efficient optimisation algorithms for inferring the parameters in complex and often high-dimensional models.

In this paper we present a framework for building such coalescence hidden Markov models for pairwise alignments and present results for using heuristic optimisation algorithms for parameter estimation. We show that we can build more complex demographic models than our previous frameworks and that we obtain more accurate parameter estimates using heuristic optimisation algorithms than when using our previous gradient based approaches.

Our new framework provides a flexible way of constructing coalescence hidden Markov models almost automatically. While estimating parameters in more complex models is still challenging we show that using heuristic optimisation algorithms we still get a fairly good accuracy.

1. Background

Coalescence theory provide a very powerful framework for genetics modelling and inference, and the coalescence process with recombination underlies many important analysis tools. Drawing inference from sequences with recombination, however, often involves integrating over all possible ancestries, modelled as the so-called ancestral recombination graph (ARG), a process that rarely scales to more than a few, short sequences due to the complexity and state space size of the ARG. To alleviate this, the *sequential Markov coalescence* approximation assumes that statistical dependencies between local genealogies are Markov (McVean and Cardin, 2005; Marjoram and Wall, 2006; Chen et al., 2009; Hobolth and Jensen, 2014).

In recent years a number of inference tools have been developed based on combining the sequential Markov coalescence with hidden Markov models, constructing so-called *coalescence hidden Markov models* or CoalHMMs, that have been constructed for the inference of speciation times (Hobolth et al., 2007; Dutheil et al., 2009; Mailund et al., 2011), gene-flow patterns (Steinrücken et al., 2013; Mailund et al., 2012), changing population sizes (Li and Durbin, 2011; Sheehan et al., 2013; Schiffels and Durbin, 2014) or inference of recombination patters (Munch et al., 2014) and have been used in a number of whole genome analyses (Locke et al., 2011; Scally et al., 2012; Prado-Martinez et al., 2013; Prüfer et al., 2012; Miller et al., 2012). These models exploit that even a very small sample of full genomic sequences holds a wealth of information about the sample's ancestry: Loci sufficiently far apart in the genome can, because of recombination in the sample's history, be considered essentially independent samples from the underlying sample populations.

The crux of constructing a CoalHMM is describing the probability of transitioning from one local genealogy along a sequence alignment to the next in terms of the underlying population genetics parameters of interest. This is typically done either by considering the probability of changing to a new genealogy conditional on a current one (Hobolth and Jensen, 2014; Li and Durbin, 2011) or by considering the joint distribution of two neighbouring trees (Dutheil et al., 2009; Mailund et al., 2011). In either case it

* Corresponding author. Tel.: +45 87155572.
E-mail addresses: ycheng@birc.au.dk (J.Y. Cheng), mailund@birc.au.dk (T. Mailund).

http://dx.doi.org/10.1016/j.compbiolchem.2015.02.001

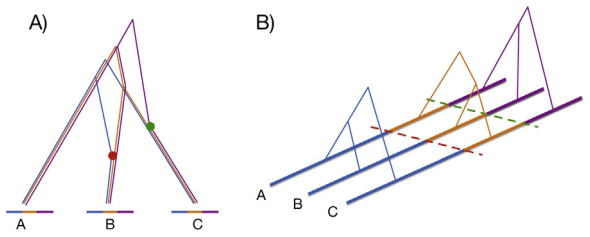

Fig. 1. (A) An ancestral recombination graph over three sequences, showing two recombinations and (B) the corresponding three local genealogies. The example shows the ancestry of three sequences in the case where they have experienced two recombination events, shown in red and green. These recombinations segments the sequences into three regions, shown in blue, orange and purple, each with different tree genealogies.

involves the explicit enumeration of all possible genealogies and a set of formulas for each possible transition. The formulas for transition probabilities, however, are very similar for transitions between similar genealogies and so constructing these formulas can be somewhat automated (Mailund et al., 2012).

Below we give a short introduction to the essentials of coalescence theory and coalescent hidden Markov models for inference of demographic parameters and in Section 3 we describe a new framework we have developed that makes it simple to construct so-called isolation-with-migration demographic models for analysis of pairwise alignments. This framework is similar to a more general framework for larger sample sizes (Mailund et al., 2012) but automates much of the model specification. The new framework is available under open source licence GPLv2 at https://github.com/mailund/IMCoalHMM.

1.1. Coalescence processes

Coalescence theory (Hein et al., 2005) describes the ancestry of a sample of present day genes and gives probabilities to all the possible genealogies that could have created the variation seen in the samples. The typical description of the coalescence model is as a continuous time Markov process running backwards in time, describing the various events that could have occurred in the past. An outcome of such a process is a tree-genealogy where inner nodes correspond to where two lineages find their most recent common ancestor. The time-depths of these nodes, and thus the branch lengths of the tree, are given by the rate of coalescence, a parameter that is determined by the size of the population the samples are taken from.

Extended with recombination, each lineage can also split into two. At a recombination event a lineage is split into a left and a right segment that then evolve back in time as two independent lineages. The outcome of this process is no longer a tree but a directed acyclic graph called the *ancestral recombination graph* or ARG (see Fig. 1A). While not a tree itself, the ARG represents a set of trees since at each position along the sample sequences a single tree describes the genealogy at that position (see Fig. 1B). At positions where a recombination has occurred the tree to the left and to the right of the recombination position can be different. The probability density over all possible ARGs thus also provides a joint probability for all the corresponding local tree-genealogies.

Structured populations can be modelled by assigning lineages to different populations, allow migration events to move lineages from one population to another, and only allow lineages to coalesce when within the same population. Population splits or admixing can be added simply by setting populations to be equal or randomly assigning lineages with one label to two or more new population labels.

Mutations on lineages can also be considered events that can occur as the process runs back in time, but typically mutations are put on the coalescence tree or ARG after it is simulated. There, the mutations can simply be put on the genealogy as a Poisson process or be put on inner nodes using a substitution model. The latter approach makes it possible to sum over all possible sequences at internal nodes using standard methods such as Felsenstein's peeling algorithm (Felsenstein, 1981) and this way obtain a joint probability distribution for the sequences at the leaves, i.e. the present day samples. This distribution depends only on the local tree-genealogies induced by the ARG since the possible nucleotides at any given position only depends on the tree for that given position.

If we denote by θ the relevant parameters for the coalescence process, e.g. coalescence rates, migration rates, recombination rates and mutation rates, we can let $f(\mathcal{G} \mid \theta)$ denote the probability density for the process producing the specific genealogy $\mathcal{G}$ and let $f(\mathcal{A} \mid \mathcal{G}, \theta)$ denote the probability that putting mutations on genealogy $\mathcal{G}$ produces the aligned samples $\mathcal{A}$. Typically the latter only depends on the mutation rate while the former is independent of the mutation rate but depends on rates (migration, recombination etc.) and time units (e.g. times where a population split apart or migration between two populations happend). These latter parameters can be expressed in time units of mutations, in essence setting $\mu = 1$, so we can simplify the two densities to just $f(\mathcal{G} \mid \theta)$ and $f(\mathcal{A} \mid \mathcal{G})$.

For demographic inference it is the parameters θ that are of interest rather than the actual underlying genealogy which is considered a nuisance parameter to be integrated out to get the likelihood

$$\mathrm{lhd}(\theta \mid \mathcal{A}) = \int f(\mathcal{A} \mid \mathcal{G}) f(\mathcal{G} \mid \theta) \ \mathrm{d}\mathcal{G}.$$

This integral over all possible genealogies is generally not efficiently computable and must either be approximated through sampling approaches or by approximating the coalescence process with a simpler model where the integral *can* be computed. The latter is the approach taken with coalescence hidden Markov models.

1.2. Coalescence hidden Markov models

The key approximation in CoalHMMs is assuming that the distribution of local genealogies along an alignment is Markov in the sense that when moving from one tree to another across a recombination point, the next tree depends only on the current tree and not any others. By approximating the distribution of local genealogies by a Markov chain the probability of the full genealogy reduces to specifying the joint probability of two neighbouring genealogies (which might be identical genealogies, e.g. if there is no recombination between them). Let ℓ denote the "left" genealogy and r the "right" genealogy and $J_\theta(\ell, r)$ their joint density. Then the "transition density" $T_\theta(r \mid \ell)$ is given simply by

$$T_\theta(r \mid \ell) = \frac{J_\theta(\ell, r)}{p_\theta(\ell)}.$$

where we define

$$p_\theta(\ell) = \int J_\theta(\ell, r') \mathrm{d}r'.$$

as the marginalisation over all possible right genealogies and thus the likelihood for just seeing the left genealogy.

If our data $\mathcal{A}$ consists of L nucleotides then the underlying genealogy $\mathcal{G}$ consists of L local trees $\mathcal{G} = \mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_L$ then

$$f(\mathcal{G} \mid \theta) = p_\theta(\mathcal{G}_1) \prod_{i=2}^{L} T_\theta(\mathcal{G}_i \mid \mathcal{G}_{i-1}).$$

The alignment probability given these local genealogies separates into probabilities for the individual nucleotides so if $\mathcal{A}_i$ denotes the i'th column in the alignment then

$$f(\mathcal{A} \mid \mathcal{G}) = \prod_{i=1}^{L} E(\mathcal{A}_i \mid \mathcal{G}_i).$$

where $E(\mathcal{A}_i \mid \mathcal{G}_i)$, the "emission probability", is the probability that the $\mathcal{A}_i$ column was produced by tree $\mathcal{G}_i$ and can be computed using the peeling algorithm.

In order to integrate over all genealogies we further approximate by discretising the possible time points where inner nodes can be found in the trees. We split the possible coalescence times into n intervals and place all events in the same interval at a single time point. This reduces the space of possible genealogies to a finite set that can be explicitly summed over, so

$$p_\theta(\ell) = \sum_{r'} J_\theta(\ell, r').$$

and

$$\int f(\mathcal{A} \mid \mathcal{G}) f(\mathcal{G} \mid \theta) \mathrm{d}\mathcal{G} = \sum_{\mathcal{G}_1, \ldots, \mathcal{G}_L} \left[p_\theta(\mathcal{G}_1) E(\mathcal{A}_1 \mid \mathcal{G}_1) \prod_{i=2}^{L} T_\theta(\mathcal{G}_i \mid G_{i-1}) E(\mathcal{A}_i \mid \mathcal{G}_i) \right].$$

This equation takes the form of a hidden Markov model (Rabiner, 1989) where the sequence $\mathcal{A}_1, \ldots, \mathcal{A}_L$ is the observable sequence and $\mathcal{G}_1, \ldots, \mathcal{G}_L$ the hidden Markov sequence. There is an exponential number of genealogies this way but by rearranging the sum and using dynamic programming in what is known as the Forward algorithm it can be computed in time $O(N^2 L)$ where L is the sequence length and N the number of possible genealogies. In the framework we describe in this paper we always consider pairwise alignments so a local genealogy consists simply of a coalescence time and with n time intervals there are n possible genealogies, and thus the likelihood of a demographic model can be computed in $O(n^2 L)$ running time using a CoalHMM, once $J_\theta(\ell, r)$ is specified.

In practise we can exploit repetitions in the alignment to reduce it further and in our framework we use the ZipHMM library (Sand et al., 2013) that lets us compute the likelihood of an entire genome alignment in a a few seconds to a few minutes depending on how finely we discretise time. For this library we simply need to specify the hidden Markov model using the transition matrix $T_\theta(r \mid \ell)$ which we compute using $J_\theta(\ell, r)$ and the emission matrix $E(\mathcal{A}_i \mid \mathcal{G}_i)$ which we compute using a Jukes-Cantor substitution model (Jukes and Cantor, 1969), where it is simply determined by the coalescence time of the $\mathcal{G}_i$ genealogy. The way our new framework makes it almost automatic to compute $J_\theta(\ell, r)$ is described in Section 2.

1.3. Parameter inference

Previous versions of our CoalHMM framework used the Nelder–Mead method (Nelder and Mead, 1965), or downhill simplex method, to estimate the parameter set for a CoalHMM by maximising the log-likelihood values calculated from the Forward algorithm. This optimisation method was developed by John Nelder and Roger Mead in 1965 as a technique to minimise an objective function in a many-dimensional space. In the context of CoalHMM, each dimension corresponds to a model parameter. CoalHMM infers parameters using maximum likelihood estimations, so the scores returned from its objective function simply correspond to the negated log-likelihood values.

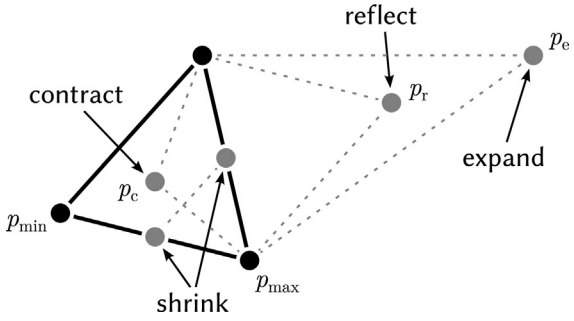

Fig. 2. An iteration of the Nelder–Mead method over two-dimensional space, showing point p_{min} reflected to point p_r, expanded to point p_e, or contracted to point p_c. If these test points do not improve the overall score of the simplex, then it shrinks around the point p_{max} with the highest score.

The Nelder–Mead method is an iterative process that continually refines a simplex, which is a polytope of $D+1$ vertices in D dimensions. During each iteration, the objective function is evaluated to determine a score at each point in the simplex (see Fig. 2). The point p_{min} with the lowest score is reflected through the centroid of the remaining vertices to point p_r. If the score at p_r is neither the highest nor the lowest score, then p_r is used in place of p_{min} to form the simplex for the next iteration. If the score at p_r is the highest score in the simplex, then this reflected point is expanded away from the centroid to p_e and used in place of p_{min} to form the next simplex. If the score at p_r is still the lowest score, then p_r is contracted toward the centroid to point p_c. If the score at p_c is no longer the lowest score, then it is used to replace p_{min} to form the next simplex. Otherwise, all points in the simplex shrink around the point p_{max} with the highest score. This process continues until the simplex collapses beyond a predetermined size, a maximum length of time expires, or a maximum number of iterations is reached.

The amount of effect these possible actions have on the simplex is controlled by supplying to the algorithm coefficients for reflection ρ, expansion χ, contraction γ, and shrinkage σ. Standard values are $\rho = 1$, $\chi = 2$, $\gamma = 1/2$, and $\sigma = 1/2$ (Baudin, 2009); but fine-tuning these coefficients has the potential to improve the performance of the algorithm.

1.3.1. Genetic algorithms

A Genetic Algorithm (GA) is a type of evolutionary algorithm. This optimisation technique gained popularity through the work of John Holland in the early 1970s (Holland, 1992). It operates by encoding potential solutions as simple chromosome-like data structures and then applying genetic alterations to those structures. Over many iterations, its population of chromosomes evolves toward better solutions, which it determines based on fitness values returned from an objective function. The algorithm typically terminates when the diversity of its population reaches a predetermined minimum, a maximum length of time expires, or a maximum number of iterations has completed.

GAs typically operate in three phases: Selection, Crossover, and Mutation (see Fig. 3). Selection determines a subset of a population what will breed the next generation of individuals, and a variety of selection schemes exist. In one scheme, Roulette Wheel Selection (RWS) (Goldberg, 1989), the algorithm selects individuals based on their relative fitness within the population; the probability p_i of selecting an individual i is given by $p_i = f_i / \sum_{j=1}^{N} f_j$, where f_i is the fitness of the individual and N is the population size. While RWS works by repeatedly sampling the population, a variation of RWS,

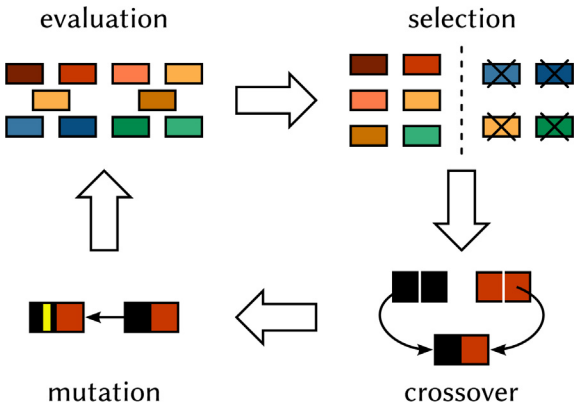

Fig. 3. In one iteration of the genetic algorithm's evolution, it operates in three stages: *Selection*, where it chooses a relatively fit subset of individuals for breeding; *Crossover*, where it recombines pairs of breeders to create a new population; and *Mutation*, where it potentially modifies portions of new chromosomes to help maintain the overall genetic diversity. Arrows in the diagram indicate transitions into the next genetic operation within one generation.

Stochastic Universal Sampling (SUS) (Baker, 1987), uses a single random value to sample all breeders by choosing them at evenly spaced intervals; this gives less fit individuals a greater chance to breed. RWS and SUS are both examples of fitness proportionate selection, but other selection schemes are based only on rank, and these are particularly beneficial when the lower and upper bounds of a fitness function are hard to determine. For example, in Tournament Selection (Miller et al., 1995), the algorithm selects an individual with the highest fitness value from a random subset of the population.

Crossover is a genetic operation used to combine pairs of individuals previously selected for breeding the following generation, and like Selection, several Crossover schemes exist. In One Point Crossover, the algorithm chooses a single point on both parents' chromosomes, and it forms the child by concatenating all data prior to that point from the first parent with all data after that point from the second parent. In Two Point Crossover, the algorithm instead chooses two points, which splits the parents' chromosomes into three regions; the algorithm then forms the child by concatenating the first region from the first parent, the second region from the second parent, and the third region from the first parent. While nature serves as the inspiration for One and Two Point Crossover, Uniform Crossover (Syswerda, 1989) has no such biological analogue. In Uniform Crossover, each position on the child's chromosome has equal opportunity to inherit its data from either parent.

Mutation is the third phase in many GAs. Every position on every chromosome has a certain probability to mutate, which helps the population maintain or even improve its genetic diversity. Several variants of this technique exist. In Uniform Mutation (Michalewicz, 1996), when a position mutates, the algorithm replaces its value with a new value, chosen at random, between a predetermined lower and upper bound. In another variant, Gaussian Mutation (Deb, 2001), when a position mutates, its current value increases or decreases based on a Gaussian random value.

1.3.2. Particle swarm optimisation

Particle Swarm Optimisation (PSO) is another type of heuristic based search algorithm. Eberhart and Kennedy first discovered and introduced this optimisation technique through simulation of a simplified social model in 1995 (Eberhart and Kennedy, 1995). Similar to GAs, PSOs are highly dependent on stochastic processes. Each individual in a PSO population maintains a position and a velocity as it flies through a hyperspace in which each dimension corresponds to one position in an encoded solution. Each individual contains a current position, which evaluates to a fitness value. Each individual also maintains its personal best position $\mathbf{p}_i$ and tracks the global best position $\mathbf{p}_g$ of the swarm (see Fig. 4). The former encapsulates the cognitive influence, and the latter encapsulates the social influence. A PSO works as an iterative process. After each iteration, the algorithm adjusts the position of each individual based on its knowledge of $\mathbf{p}_i$ and $\mathbf{p}_g$. This adjustment is analogous to the crossover operation used by GAs. The inertia of an individual, however, allows it to overshoot local minima and explore unknown regions of the problem domain.

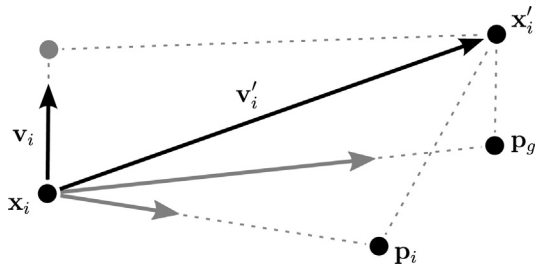

Fig. 4. Three vectors applied to a particle at position $\mathbf{x}_i$ in one iteration of a Particle Swarm Optimisation: a cognitive influence urges the particle toward its previous best $\mathbf{p}_i$, a social influence urges the particle toward the swarm's previous best $\mathbf{p}_g$, and its own velocity $\mathbf{v}_i$ provides inertia, allowing it to overshoot local minima and explore unknown regions of the problem domain.

In PSO, we represent the position of the ith particle as $\mathbf{x}_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,D})$ and its velocity as $\mathbf{v}_i = (v_{i,1}, v_{i,2}, \ldots, v_{i,D})$, where D is the number of dimensions in the parameter space. We represent the particle's previous position with its best fitness as $\mathbf{p}_i = (p_{i,1}, p_{i,2}, \ldots, p_{i,D})$. During each iteration, the algorithm adjusts the velocity $\mathbf{v}$ and position $\mathbf{x}$ according to the following equations:

$$\begin{aligned}\mathbf{v}'_{i,d} \leftarrow \mathbf{v}_{i,d} \quad & +\phi_p \cdot r_p \cdot (\mathbf{p}_{i,d} - \mathbf{x}_{i,d}) \\ & +\phi_g \cdot r_g \cdot (\mathbf{p}_{g,d} - \mathbf{x}_{i,d}) \\ \mathbf{x}'_{i,d} \leftarrow \mathbf{x}_{i,d} \quad & +\mathbf{v}_{i,d}.\end{aligned}$$

where r_p and r_g are two random values between zero and one, and ϕ_p and ϕ_g are two positive constants representing cognitive and social influences. As Shi and Eberhart demonstrated (Shi and Eberhart, 1998), it can be beneficial to include a constant ω, which helps balance the global and local search forces. This term directly affects the inertia of the particle.

$$\begin{aligned}\mathbf{v}'_{i,d} \leftarrow \omega \cdot \mathbf{v}_{i,d} \quad & +\phi_p \cdot r_p \cdot (\mathbf{p}_{i,d} - \mathbf{x}_{i,d}) \\ & +\phi_g \cdot r_g \cdot (\mathbf{p}_{g,d} - \mathbf{x}_{i,d}).\end{aligned}$$

2. Methods

We first describe how our framework supports constructing CoalHMMs for pairwise alignments and then the algorithms we have implemented for parameter estimation.

2.1. Framework for CoalHMMs for pairwise alignments

Our framework builds the joint probability distribution $J_\theta(\ell, r)$ by tracking all possible states of the coalescence process for two samples with two nucleotides similar to our previous work (Mailund et al., 2011, 2012, 2012). Demographic scenarios are specified by slicing the past into a number of "epochs" where each such has a fixed number of populations and a fixed number of constant rates with which events occur. Within each epoch we construct the state space of all possible configurations within the demographic model of the epoch and construct a continuous time Markov chain (CTMC). Finally we stack these CTMCs on top of each other to get

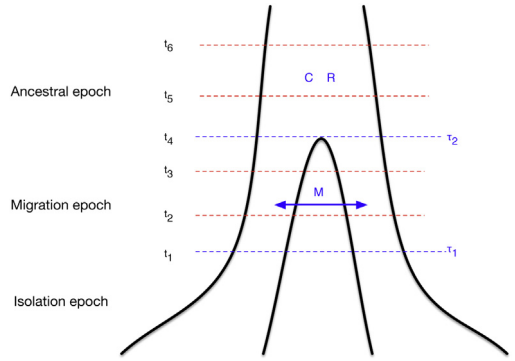

Fig. 5. The demographic IIM model. The model has three epochs and five parameters. An ancestral population epoch with one population and free coalescences, a migration epoch with two populations where lineages can only coalesce within the same population but can migrate between the populations, and an isolation epoch where the two populations are completely independent. The parameters are the time points where the system switches between the epochs, the coalescence and recombination rates (assumed to be the same in all populations) and a symmetric migration rate during the migration epoch. The time point $t_1, t_2, \ldots, t_6$ illustrates a possible discetisation of time into the intervals that becomes the states of the hidden Markov model.

a coalescence process for the two samples for all the combined epochs and from this compute the joint probabilities of which intervals the left and right nucleotides will coalesce in.

As an example, consider the Isolation with Initial Migration (IIM) model from Mailund et al. (2012) and shown in Fig. 5. This model has three epochs. From the most recent to the most ancient these are (1) and epoch with complete isolation where lineages in the two populations can never coalesce, (2) an epoch with population structure where there are two distinct populations but where lineages can migration between them, and finally (3) an epoch with a single ancestral population.

The first epoch allows lineages to recombine and coalesce within each population but does not allow migrations. In this time period it is not possible for the two samples to find a common ancestor. In the migration period, lineages can cross from one population to another and coalesce into a common ancestor. In the final epoch the lineages can coalesce and find common ancestors freely. To build a CoalHMM for this demographic model it is necessary to build CTMCs for the three epochs, combine them to build a model for the entire demographic past and then use this model to specify the joint probability $J_\theta(\ell, r)$.

2.1.1. Building continuous time Markov chains

To track the possible histories within an epoch we explicitly construct the state space of the two-locus coalescence process; an approach taken in several earlier papers (Slatkin and Pollack, 2006; Simonsen and Churchill, 1997; Mailund et al., 2011; Hobolth and Jensen, 2014). Since explicitly enumerating all states and transitions is both tedious and error-prone we avoid this by letting the computer enumerate all states in a transition system. The states and transitions are defined as in our previous IIM paper (Mailund et al., 2012) but repeated below for completeness of this paper.

We represent lineages at a single nucleotide as sets. The sets $\{1\}$ and $\{2\}$ denote sequences 1 and 2 before they have found a common ancestor while $\{1,2\}$ denote a lineage ancestral to both. We then model two neighbouring nucleotides as pairs of such states, so e.g. $(\{1,2\},\{1\})$ denote a lineage where the left nucleotide has found a common ancestor between sample 1 and 2 and is linked on the right to a nucleotide from the sequence 1, which has not found a common ancestor with sequence 2. To assign lineages to species, we pair them again, and let $[1,(l,r)]$ denote that lineage

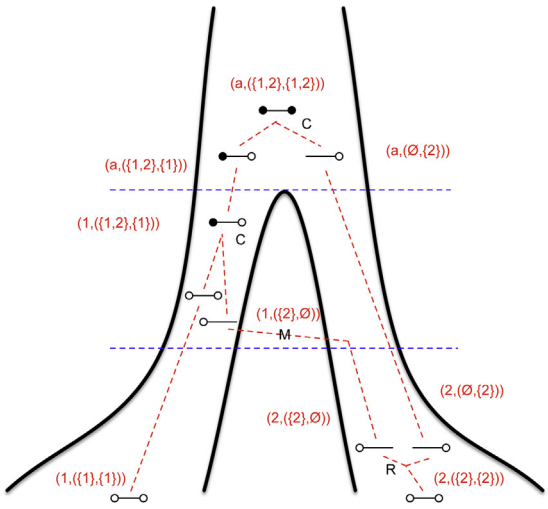

Fig. 6. An ancestral recombination graph in the IIM model with lineages in the notation of the transition system. The state at any particular point in time, corresponding to a horizontal line through the ARG, would be the number of lineages at that particular time. The initial state is {(1,({1},{1})), (2,({2},{2}))} that through a recombination transition (R) moves to {(1,({1},{1})), (2,({2},Ø)), (2,(Ø,{2}))}. The system now moves from its isolation epoch to its migration epoch and the next event is a migration event (M) that changes the state to {(1,({1},{1})), (1,({2},Ø)), (2,(Ø,{2}))} followed by a coalescence event (C) and the state {(1,({1,2},{1})), (2,(Ø,{2}))}. Now the system moves to the ancestral population epoch where this state is projected to the state {(*a*, ({1, 2}, {1})), (*a*,(Ø,{2}))} and the final event is a coalescence event changing the state to {(*a*, ({1, 2}, {1, 2}))}.

(l, r) is in population 1. A state in the CTMC corresponds to a set of such lineages assigned to species.

We define the following transitions of states:

$$\text{Coalescence: } \{[p_1,(l_1,r_1)]\} \cup \{[p_2,(l_2,r_2)]\} \cup \mathcal{S} \to \{[p_1,(l_1 \cup l_2, r_1 \cup r_2)]\} \cup \mathcal{S} \text{ if } p_1 = p_2$$

$$\text{Recombination: } \{[p,(l,r)]\} \cup \mathcal{S} \to \{[p,(l,\emptyset)]\} \cup \{[p,(\emptyset,r)]\} \cup \mathcal{S}$$

$$\text{Migration: } \{[p_1,(l,r)]\} \cup \mathcal{S} \to \{[p_2,(l,r)]\} \cup \mathcal{S} \text{ if } p_1 \neq p_2.$$

where $\mathcal{S}$ denotes the set of other lineages in the state.

When migration is not allowed in the epoch, as in the first epoch in the IIM model, we simply leave that transition out of the transition system when computing the state space. Fig. 6 shows and example of a run in this transition system specified for the IIM model.

As the initial state of the system, we use the state where sequence 1 is in population 1, sequence 2 is in population 2, and both sequences have their left and right nucleotides linked, $\{[1,(\{1\},\{1\})],[2,(\{2\},\{2\})]\}$, and we then compute a graph of all states reachable from this state through the transitions above, labelling each edge with the kind of transformation (coalescence, recombination or migration). From this state space we construct a rate matrix for the CTMC by first assigning a number to each state, and then setting rates (from our parameters θ) in the matrix in entries corresponding to edges in the graph. This is translated into an instantaneous rate matrix for the CTMC by setting all diagonal cells to minus the row sum. The result is a rate matrix for the CTMC, Q, such that $Q_{x,y}$ is the instantaneous rate of moving from state x to state y. From CTMC theory the probability of moving from state x to state y in time t is then given by $\left(e^{Qt}\right)_{x,y}$ where e^{Qt} is matrix exponentiation (Moler and Van Loan, 2003). For each time interval i in the CoalHMM we let Q^i denote the rate matrix of that interval. For intervals in the same epoch these will of course share the rate matrix but not necessarily the probability matrix for moving from one state to another when going through

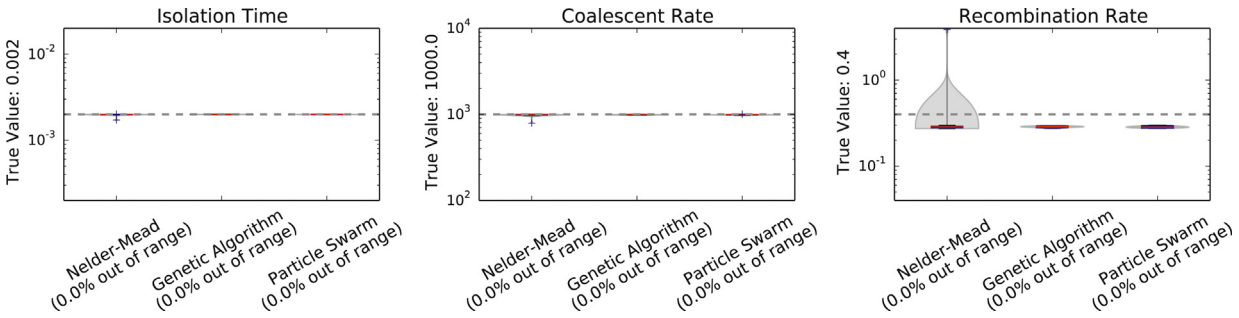

Fig. 7. Estimates for the isolation model from three optimisation algorithms. All three optimisers recover the simulated parameters, shown as dashed horizontal lines, reasonably well but with a higher variance for the Nelder–Mead optimiser. The estimates of the recombination rate are downwards biased, an effect we have previously observed and speculate is a consequence of the Markov assumption (Mailund et al., 2011).

the interval since the intervals do not necessarily have the same length.

2.1.2. Computing joint probabilities

To compute the $J_\theta(\ell, r)$ probabilities we use ideas from Mailund et al. (2011). Since coalescence times are discretised in time intervals we use $J_\theta(\ell = i, r = j)$ to mean that the left nucleotide coalesced in interval i and the right nucleotide in interval j. For this to be the case, and assuming interval i is earlier than interval j, neither left nor right nucleotide can have found a common ancestor between the two samples when entering interval i, the left but only the left must have when leaving interval i and this must remain the case until entering interval j, and when leaving interval j both left and right nucleotides must have found common ancestors for the two samples.

Regardless of the state space for the epoch CTMC we can always split the states into four non-overlapping (but possibly empty) sets: B: the "beginning states" where neither nucleotides have found common ancestors, L: the "left states" where the left nucleotides but not the right nucleotides have found a common ancestor, R: the "right states" where the right nucleotides but not the left nucleotides have found a common ancestor, and E: the "end states" where both nucleotides have found common ancestors. In terms of these sets we can reformulate the conditions for $J_\theta(\ell = i, r = j)$ as follows: when entering interval i we must be in a state in B but when leaving interval i we must be in a state in L and we must remain in L until we enter interval j and leave interval j in a state in E. It is straightforward to identify which of these sets each state belongs to and our framework does this automatically regardless of the epochs specification. We will use sub-scripts to indicate which interval and thus epoch the sets are associated with, so B_i, L_i, R_i and E_i are the sets for interval i.

Let $\mathcal{T}^i$ denote the probability transition matrix for changing states when going through interval i as computed from the matrix exponentiation of the rate matrix for the epoch of the interval $\mathcal{T}^i = e^{Q^i \Delta t_i}$ where Δt_i is the length of interval i. Since the state space in interval i and interval $i+1$ are not necessarily the same – if the intervals are from different epochs they might not be – we use the convention that the rows of $\mathcal{T}^i$ are indexed with the state space for interval i and the columns with the state space for interval $i+1$; the starting states for $\mathcal{T}^i$ are from the CTMC for interval i but the end states are from the CTMC for interval $i+1$. This makes it possible to always multiply together $\mathcal{T}$ matrices from adjacent intervals.

When two adjacent intervals are from the same epoch, then $\mathcal{T}^i$ is specified just from the matrix exponentiation, but when the interval i is from one epoch and $i+1$ from another, with a different state

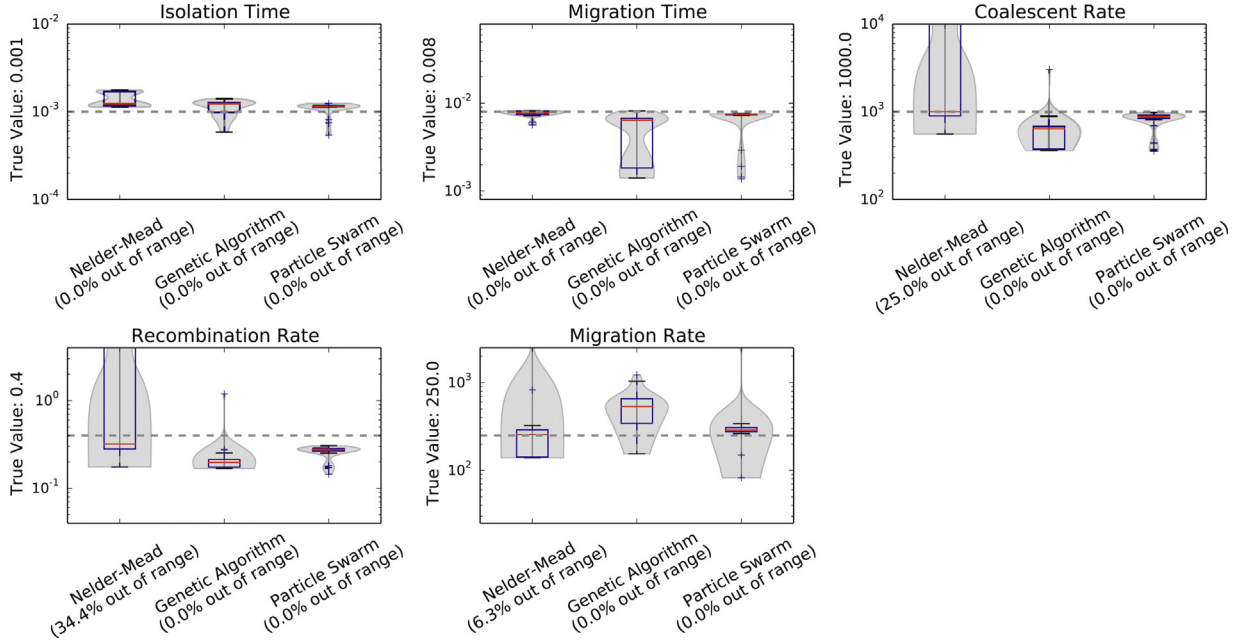

Fig. 8. Estimates for the isolation with initial migration model from all three optimisation algorithms. With more parameters to estimate, the variance in the estimates goes up as expected. The parameters are still reasonably well estimated for the two heuristic optimisers, especially for the particle swarm optimiser, but less so for the Nelder–Mead optimiser.

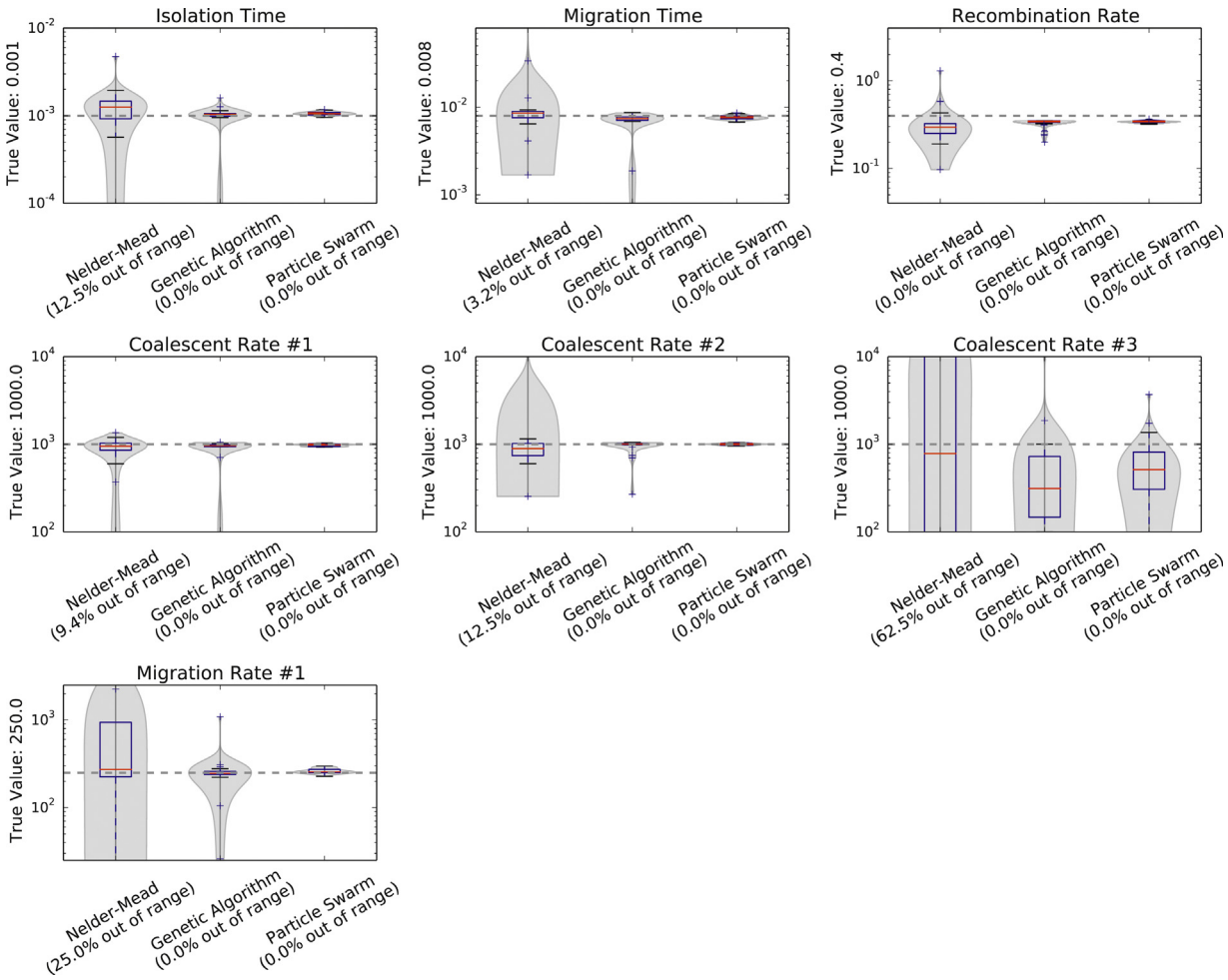

Fig. 9. Estimates for the isolation with initial migration model three epochs: One isolation epoch, one migration epoch and one ancestral epoch. This corresponds to the IIM model except that the coalescence rate is not assumed to be the same in all epochs. Again we see a failure for the Nelder–Mead to recover these parameters, and the last coalescence rate is not well estimated. The particle swarm optimiser performs the best among three optimisers.

space, a projection matrix is necessary. Such a matrix specifies how states in one CTMC correspond to states in another and by placing 1s in the relevant entries in a matrix P the $\mathcal{T}^i$ matrix is computed simply as $\mathcal{T}^i = e^{Q\Delta t_i} \cdot P$. In the case of the IIM model, moving from the isolation epoch to the migration epoch, lineages are mapped directly as $[p_i, (l, r)] \mapsto [p_i, (l, r)]$ since the lineages in the individual populations are the same; the state space is just larger when migration is allowed. For going from the migration epoch into the ancestral population both population p_1 and p_2 are simply mapped to the ancestral population p_A: $[p_i, (l, r)] \mapsto [p_A, (l, r)]$. We refer to the documentation in the framework for details on this and more complex projections.

Let $\mathcal{U}^i$ denote the transition matrix for going from time zero until the start point of interval i. This can be computed from the $\mathcal{U}^1$ matrix for getting from time zero to the first interval and $\mathcal{T}^j$ matrices for $j < i$:[1]

$$\mathcal{U}^i = \mathcal{U}^1 \prod_{j=1}^{i-1} \mathcal{T}^j.$$

If the first interval starts at time zero, $\mathcal{U}^1$ will just be the identity matrix. If it is not possible to coalesce for a certain time, as in the IIM model where the lineages are isolated until migration becomes possible, then the first interval starts later than time zero and $\mathcal{U}^1$ is used to address this. In the IIM $\mathcal{U}^1$ is computed by exponentiating the rate matrix from the isolation model multiplied with the isolation time.

Finally, let $\mathcal{B}^{i,j}$ for $i < j$ denote the probability matrix for going from the beginning of interval i to the end of interval j. This can be computed as

$$\mathcal{B}^{i,j} = \prod_{k=i}^{j} \mathcal{T}^k.$$

For computing $J_\theta(\ell = i, r = j)$ there are three cases: $i = j$, $i < j$ and $i > j$. All can be computed using the matrices defined above. Let ι denote the initial state for the coalescence system at time zero. For the IIM this would be the two lineages in separate populations. For $i = j$ we have

$$J_\theta(\ell = i, r = i) = \sum_{b \in B_i} \sum_{e \in E_{i+1}} \mathcal{U}^i_{\iota,b} \cdot \mathcal{T}^i_{b,e}.$$

with a special case for the last interval

$$J_\theta(\ell = n, r = n) = \sum_{b \in B_n} \mathcal{U}^n_{\iota,b}.$$

[1] In the actual implementation, intervals are indexed from zero and $\mathcal{U}^1$ is called $\mathcal{U}^0$ but we have chosen to index from 1 in the explanation of the algorithm here.

For $i < j$ we have

$$J_\theta(\ell = i, r = j) = \sum_{b \in E_i} \sum_{l \in L_{i+1}} \sum_{l' \in L_j} \sum_{e \in E_{j+1}} \mathcal{U}^i_{\iota,b} \cdot \mathcal{T}^i_{b,l} \cdot \mathcal{B}^{i+1,j-1}_{l,l'} \cdot \mathcal{T}^j_{l',e}.$$

with again a special case for the last interval

$$J_\theta(\ell = i, r = n) = \sum_{b \in E_i} \sum_{l \in L_{i+1}} \sum_{l' \in L_n} \mathcal{U}^i_{\iota,b} \cdot \mathcal{T}^i_{b,l} \cdot \mathcal{B}^{i+1,n-1}_{l,l'}.$$

Since the coalescence process is symmetric in left and right we can simply compute the cases for $j < i$ as $J_\theta(\ell = i, r = j) = J_\theta(\ell = j, r = i)$.

To specify a CoalHMM in our framework it is only necessary to specify the $\mathcal{T}^i$ and $\mathcal{U}^1$ matrices. Mostly this is a simple case of exponentiating rate matrices and specifying projections when moving between epochs.

2.2. Optimisation algorithms

We have enhanced our framework by incorporating two heuristic based optimisation algorithms. In both algorithms, the fitness of an individual solution is the negated log-likelihood values computed from the Forward algorithm from the CoalHMM.

2.2.1. Genetic algorithm optimiser

A GA optimiser in the CoalHMM framework initiates its first generation of individuals by uniformly selecting parameters within predetermined ranges. The GAs use population sizes of 100. Small

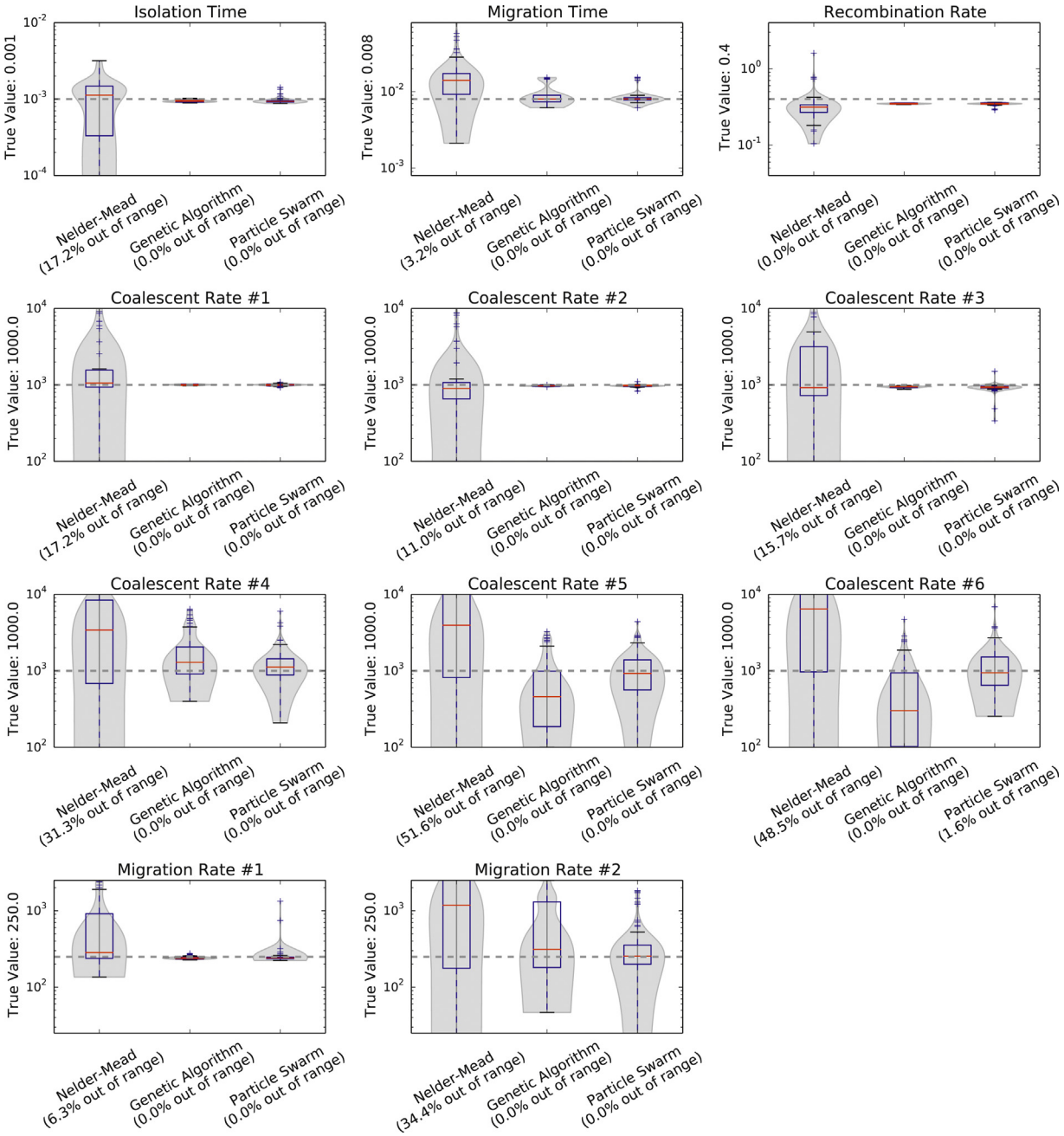

Fig. 10. Estimates for the isolation with initial migration model six epochs. Results are similar to the three epochs model. We see a failure to recover most of the parameters from the Nelder–Mead. We see some suboptimal results for the last two coalescence rates and second migration rate from the genetic algorithm. We see the best accuracy from the particle swarm.

populations lose genetic diversity quickly, while large populations result in better accuracy at the cost of increased running time. For our models, population sizes greater than 100 did not offer significant improvement. To form the breeding pool, we use Tournament Selection with a selection rate of 75% of the population size with tournament sizes of 10. We use a rank-based selection scheme because the lower and upper bounds of the fitness are unknown beforehand and differ from model to model; in order to use fitness proportionate selection, we would need an initial phase to estimate the fitness range. We then use One Point Crossover to

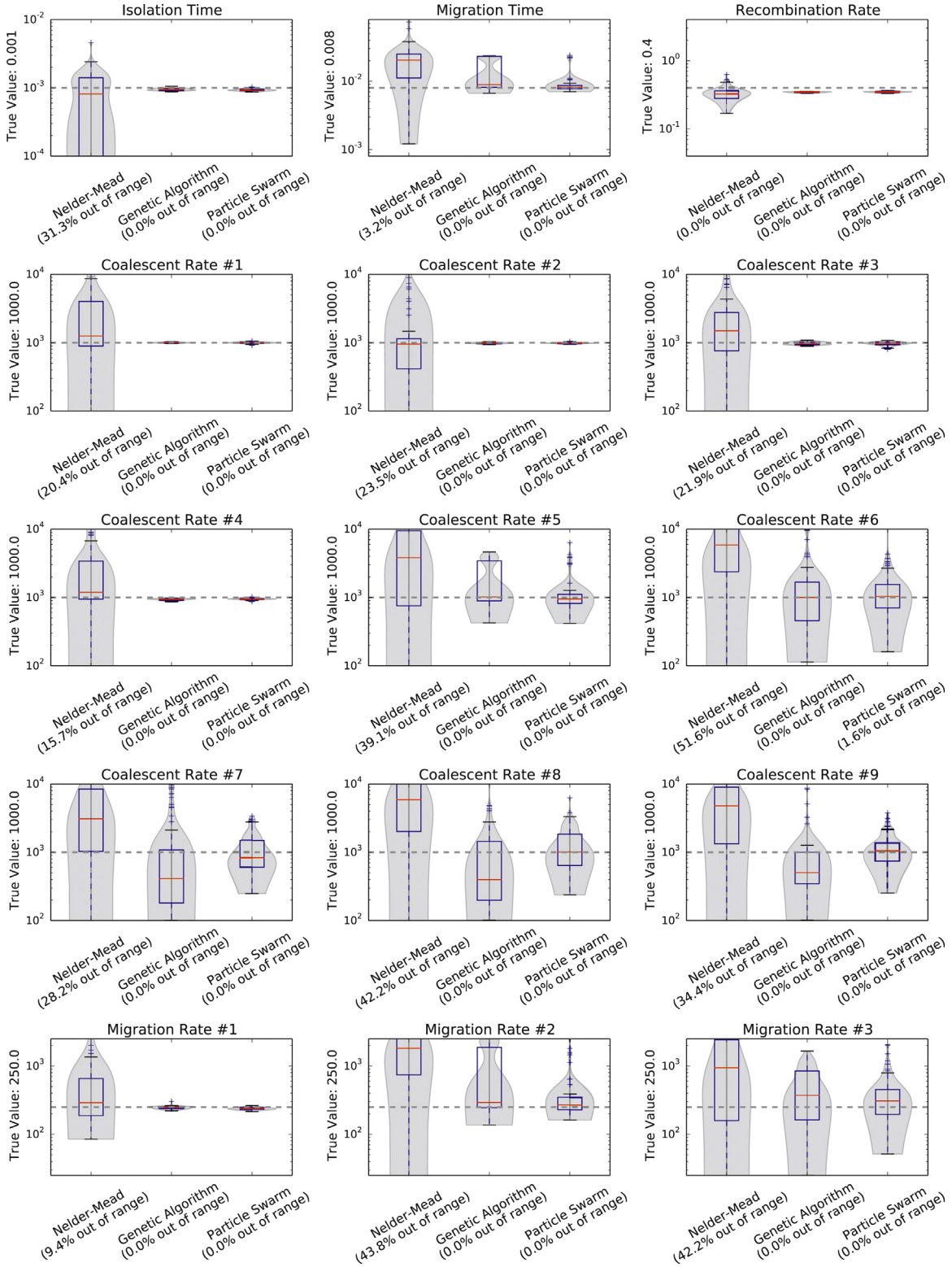

Fig. 11. Estimates for the isolation with initial migration model nine epochs. Results are similar to the three epochs and the six epochs models. We see a failure to recover most of the parameters from the Nelder–Mead. We see some suboptimal results for the late coalescence rates and migration rates from the genetic algorithm. We see the best accuracy from the particle swarm.

combine two breeders and generate individuals for the next generation. We chose this simple crossover scheme because other complex schemes failed to produce improved results. To help genetic diversity in a population, we apply point mutations at a rate of 15% and use Gaussian Mutation with $\mathcal{N}(\mu = 0, \sigma = 0.01)$. This relatively high point mutation rate is balanced by the relatively low σ; this configuration is suitable for our problem space, which consists of short chromosomes encoded with real numbers.

2.2.2. *Particle swarm optimiser*

Our framework also provides a PSO optimiser. Each model parameter corresponds to a dimension in the solution space. The

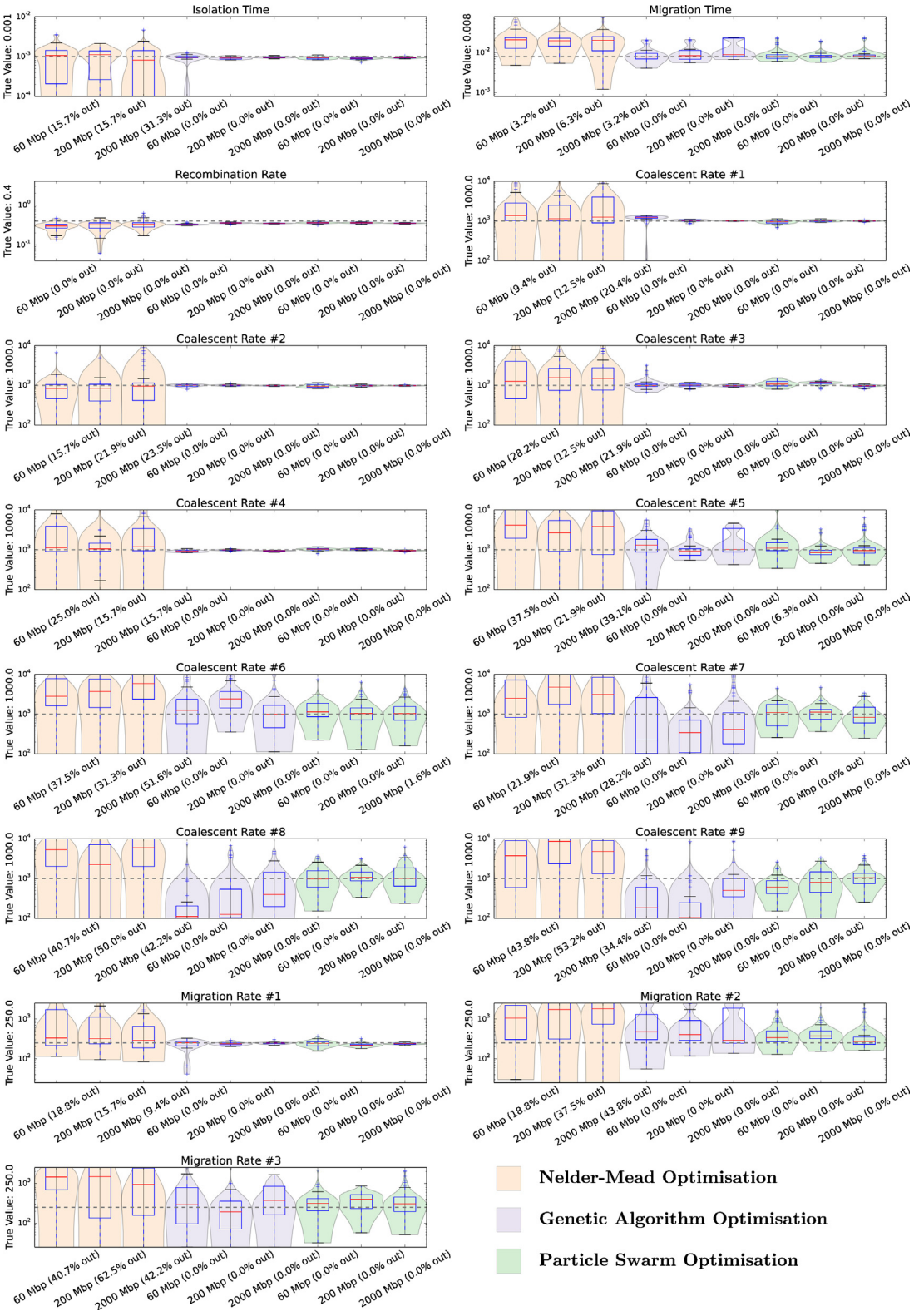

Fig. 12. Estimation accuracy with variable data sizes. For some of the parameters we see a reduction in the estimation variance with more data, but less than one would have hoped for a factor of more than three increase in alignment length.

optimiser initialises particle velocities from uniform random values within a range of 2% of the predetermined range for each parameter. During each iteration, we update the velocities of each particle using coefficients determined from trial and error. For the inertial coefficient, we use $\omega = 0.9$; i.e. a 10% decay in velocity if the particle is not affected by other forces. For the cognitive and social coefficients, we use $\phi_p = 0.3$ and $\phi_g = 0.1$, respectively. Larger values for ϕ had the tendency to accelerate the particles beyond acceptable ranges. Similar to our GA performance, we found population sizes greater than 100 did not significantly improve the performance, but they did dramatically increase the time required for the swarm to converge.

2.3. Simulated data

We use the program **ms** to generate ancestral recombination graphs under standard neutral evolutionary models with recombination, speciation, variable populations, migrations, etc. We then use the **seq-gen** program to produce sequence samples of length 10 Mbp. Using the phylogenetic trees simulated by **ms** as input, **seq-gen** evolves the sequences along the phylogeny.

3. Results and discussion

Below we illustrate how demographic inference can be done using our new CoalHMM framework by presenting a number of demographic models, from simple to more complex, and show how we can estimate parameters using our heuristic optimisation algorithms. All models are available as inference scripts in the framework.

3.1. Isolation model

The simplest model we will consider is the clean isolation model from Mailund et al. (2011). The model has three parameters: the split time where the ancestral population is split into two independent populations, a coalescence rate that is the same for the ancestral population and the two descedent populations, and a recombination rate.

Fig. 7 shows the estimation results for all three optimisers, operating on simulated sequences consisting of 1000 Mbp. The range on the y-axis corresponds to the range of possible values for the GA and PSO optimisers for each parameter. The Nelder–Mead optimiser is not limited to these ranges and the percentage of estimates that falls outside of the range is written below the x-axis. For this simplest model all three optimisers recover the simulated parameters, shown as dashed horizontal lines, reasonably well but with a higher variance for the Nelder–Mead optimiser. The estimates of the recombination rate are downwards biased, an effect we have previously observed and speculate is a consequence of the Markov assumption (see Mailund et al. (2011) Supplemental Text 1 and Fig. 4S in the same text).

3.2. Isolation with initial migration model

The next model we consider is the IIM model from Mailund et al. (2012) that we have used as an example in Section 2. This model has five parameters: The time period where the two populations are completely isolated, the time period where migration is ongoing, a shared coalescence rate for all populations, a migration rate for the migration epoch, and a recombination rate.

Fig. 8 shows the estimation results for this model for our three optimisers, operating on simulated sequences consisting of 1000 Mbp. With more parameters to estimate, the variance in the estimates goes up as expected. The parameters are still reasonably well estimated for the two heuristic optimisers, especially for the Particle Swarm optimiser, but less so for the Nelder–Mead optimiser. We still see a bias in the estimates of the recombination rate, but now also an slight upwards bias in the estimates of the split time (the time where gene flow finally ends). This was not obvious in our previous results (Mailund et al., 2012) because of the large variance in the optimiser we used there.

3.3. Multi-epochs isolation with initial migration models

For a more complex model we consider an extension of the IIM model not previously described. This model allows multiple epochs within the isolation period, the migration period and the ancestral population. Both coalescence rates and migration rates can vary freely between epochs. In our experiments we always have the same number of isolation, migration and ancestral epochs. The parameters are the end of gene flow (split time), the beginning of gene flow (migration time), one coalescence rate for each of the isolation, migration and ancestral epochs, a symmetric migration rate for each migration epoch and the recombination rate.

The first coalescence rate would be impossible to estimate with just a pairwise alignment of one sequence from each population since we observe no coalescence events there and so would have no hidden Markov model states in that epoch (Mailund et al., 2011). We solve this by constructing a composite likelihood from three different hidden Markov models: one where our pairwise alignment is from two samples from the first population, one where the alignment is from the second alignment and one with one sample from each population. These are all constructed with the same CTMCs and only differ in the initial state, ι, used for calculating the joint genealogy probability. We run all three models in parallel with the same parameters and add the log-likelihoods together to get a combined likelihood.

Fig. 9 shows results for a model with three epochs, operating on simulated sequences consisting of 2000 Mbp. This corresponds to the IIM model except that there are now three coalescence rates instead of one. Again we see a failure for the Nelder–Mead to recover these parameters. The the last coalescence rate is not as well estimated. The particle swarm optimiser performs the best among three optimisers.

Figs. 10 and 11 show results for models with six and nine epochs, respectively, operating on simulated sequences consisting of 2000 Mbp. Results are similar to the three epochs model. We see a failure to recover most of the parameters from the Nelder–Mead and some suboptimal results for the last coalescence rates and migration rates from the Genetic Algorithm. We see a better accuracy from the Particle Swarm. Even for the nine epochs model the Particle Swarm estimates reasonably well. The earlier migration rates are estimated better than last migration rate in both the six epochs model and the nine epochs model.

Fig. 12 shows the effect of increasing the data size from 60 Mbp to 2000 Gbp. For some of the parameters we see a reduction in the estimation variance with more data, but less than one would have hoped for a factor of more than three increase in alignment length.

4. Conclusions and future work

We have described a new framework for constructing coalescence hidden Markov models for demographic inference and showed that using heuristic optimisation algorithms we can accurately estimate parameters in a number of complex models. Using our framework it is relatively easy to construct CoalHMMs for even rather complex demographics, but a limiting factor is the accurate parameter estimation. We have shown that the Nelder-Mead algorithm we have previously used for estimation fails somewhat when the number of parameters increases and that the heuristic

optimisers do a better job. Good optimisation algorithms is still a topic for future work.

In this paper we have focused on maximum likelihood estimates of each parameter but not considered estimating error bars for the estimates. These can be computed using bootstrap or jackknife approaches but this comes at a cost in running time. Here, as well, future work is needed.

Being able to work with larger sample sizes than four could potentially improve the accuracy of parameter estimates as shown in the MSMC (Schiffels and Durbin, 2014) model compared to the PSMC model (Li and Durbin, 2011), and some of the approaches we take in our framework generalises to more samples. The construction of CTMCs for more samples is immediately possible as we have shown in previous work (Mailund et al., 2012), although this approach will only scale to a small number of samples due to the problem of dealing with very large state spaces for the CTMCs. Automatically combining CTMCs for such cases in a similar way to what we have presented here is more complex still and requires more work.

Despite these limitations we believe that our new framework will enable more complex models to be explored using the CoalHMM methodology and that the ideas underlying its design can be used for improved frameworks in the future.

Competing interests

The authors declare that they have no competing interests.

Authors' contributions

TM implemented the CoalHMM framework. JC implemented the optimisation algorithms. Both authors designed the experiments and analysed the results. JC executed the experiments. Both authors drafted the manuscript.

Acknowledgements

This research was funded by the Danish Council of Independent Research Sapere Aude Grant 12-125062.

References

Baker, J.E.,1987. Reducing bias and inefficiency in the selection algorithm. In: Proceedings of the Second International Conference on Genetic Algorithms on Genetic Algorithms and Their Application. L. Erlbaum Associates Inc., Hillsdale, NJ, USA, pp. 14–21 http://portal.acm.org/citation.cfm?id=42512.42515

Baudin, M., 2009. Nelder Mead User's Manual.

Chen, G.K., Marjoram, P., Wall, J.D., 2009. Fast and flexible simulation of dna sequence data. Genome Res. 19 (1), 136–142.

Deb, K., 2001. Multi-objective optimization using evolutionary algorithms.

Dutheil, J.Y., Ganapathy, G., Hobolth, A., Mailund, T., Uyenoyama, M.K., Schierup, M.H., 2009. Ancestral population genomics: the coalescent hidden Markov model approach. Genetics 183 (1), 259–274.

Eberhart, R., Kennedy, J., 1995. A new optimizer using particle swarm theory. In: Proceedings of the Sixth International Symposium on Micro Machine and Human Science, 1995, pp. 39–43, http://dx.doi.org/10.1109/MHS.1995.494215.

Felsenstein, J., 1981. Evolutionary trees from DNA sequences: a maximum likelihood approach. J. Mol. Evol. 17 (6), 368–376, http://dx.doi.org/10.1007/BF01734359.

Goldberg, D.E., 1989. Genetic Algorithms in Search, Optimization and Machine Learning, 1st ed. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Hein, J., Schierup, M.H., Wiuf, C., 2005. Gene Genealogies, Variation and Evolution: A Primer in Coalescent Theory. Oxford University Press, USA.

Hobolth, A., Jensen, J.L., 2014. Markovian approximation to the finite loci coalescent with recombination along multiple sequences. Theor. Popul. Biol. 98, 48–58.

Hobolth, A., Christensen, O.F., Mailund, T., Schierup, M.H., 2007. Genomic relationships and speciation times of human, chimpanzee, and gorilla inferred from a coalescent hidden Markov model. PLoS Genet. 3 (2), e7.

Holland, J.H., 1992. Genetic algorithms. Sci. Am. 267 (1), 66–72.

Jukes, T.H., Cantor, C.R., 1969. Evolution of protein molecules. In: Munro, M.N. (Ed.), Mammalian Protein metabolism, Vol. III. Academic Press, New York, pp. 21–132.

Li, H., Durbin, R., 2011. Inference of human population history from individual whole-genome sequences. Nature 475 (7357), 493–496.

Locke, D.P., Hillier, L.W., Warren, W.C., Worley, K.C., Nazareth, L.V., Muzny, D.M., Yang, S.-P., Wang, Z., Chinwalla, A.T., Minx, P., Mitreva, M., Cook, L., Delehaunty, K.D., Fronick, C., Schmidt, H., Fulton, L.A., Fulton, R.S., Nelson, J.O., Magrini, V., Pohl, C., Graves, T.A., Markovic, C., Cree, A., Dinh, H.H., Hume, J., Kovar, C.L., Fowler, G.R., Lunter, G., Meader, S., Heger, A., Ponting, C.P., Marques-Bonet, T., Alkan, C., Chen, L., Cheng, Z., Kidd, J.M., Eichler, E.E., White, S., Searle, S., Vilella, A.J., Chen, Y., Flicek, P., Ma, J., Raney, B.J., Suh, B.B., Burhans, R., Herrero, J., Haussler, D., Faria, R., Fernando, O., Darré, F., Farré, D., Gazave, E., Oliva, M., Navarro, A., Roberto, R., Capozzi, O., Archidiacono, N., della Valle, G., Purgato, S., Rocchi, M., Konkel, M.K., Walker, J.A., Ullmer, B., Batzer, M.A., Smit, A.F.A., Hubley, R., Casola, C., Schrider, D.R., Hahn, M.W., Quesada, V., Puente, X.S., Ordo nez, G.R., López-Otín, C., Vinar, T., Brejova, B., Ratan, A., Harris, R.S., Miller, W., Kosiol, C., Lawson, H.A., Taliwal, V., Martins, A.L., Siepel, A., RoyChoudhury, A., Ma, X., Degenhardt, J., Bustamante, C.D., Gutenkunst, R.N., Mailund, T., Dutheil, J.Y., Hobolth, A., Schierup, M.H., Ryder, O.A., Yoshinaga, Y., de Jong, P.J., Weinstock, G.M., Rogers, J., Mardis, E.R., Gibbs, R.A., Wilson, R.K., 2011. Comparative and demographic analysis of orang-utan genomes. Nature 469 (7331), 529–533.

Mailund, T., Dutheil, J.Y., Hobolth, A., Lunter, G., Schierup, M.H., 2011. Estimating divergence time and ancestral effective population size of Bornean and Sumatran orangutan subspecies using a coalescent hidden Markov model. PLoS Genet. 7 (3), e1001319.

Mailund, T., Halager, A.E., Westergaard, M., Dutheil, J.Y., Munch, K., Andersen, L.N., Lunter, G., Prüfer, K., Scally, A., Hobolth, A., Schierup, M.H., 2012. A new isolation with migration model along complete genomes infers very different divergence processes among closely related great ape species. PLoS Genet. 8 (12), e1003125.

Mailund, T., Halager, A., Westergaard, M., 2012. Using colored Petri nets to construct coalescent hidden Markov models: Automatic translation from demographic specifications to efficient inference methods. In: Haddad, S., Pomello, L. (Eds.), Application and Theory of Petri Nets. Bioinformatics Research Center, Aarhus University/Springer, Denmark/Berlin, Heidelberg, pp. 32–50.

Marjoram, P., Wall, J.D., 2006. Fast "coalescent" simulation. BMC Genet. 7, 16.

McVean, G., Cardin, N.J., 2005. Approximating the coalescent with recombination, Philosophical transactions of the Royal Society of London. Series B. Biol. Sci. 360 (1459), 1387–1393.

Michalewicz, Z., 1996. Genetic algorithms -f data structures = evolution programs.

Miller, B.L., Miller, B.L., Goldberg, D.E., Goldberg, D.E., 1995. Genetic algorithms, tournament selection, and the effects of noise. Complex Syst. 9, 193–212.

Miller, W., Schuster, S.C., Welch, A.J., Ratan, A., Bedoya-Reina, O.C., Zhao, F., Kim, H.L., Burhans, R.C., Drautz, D.I., Wittekindt, N.E., Tomsho, L.P., Ibarra-Laclette, E., Herrera-Estrella, L., Peacock, E., Farley, S., Sage, G.K., Rode, K., Obbard, M., Montiel, R., Bachmann, L., Ingolfsson, Ó., Aars, J., Mailund, T., Wiig, Ø., Talbot, S.L., Lindqvist, C., 2012. Polar and brown bear genomes reveal ancient admixture and demographic footprints of past climate change. PNAS 109 (36), E2382–E2390.

Moler, C., Van Loan, C., 2003. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. SIAM Rev. 45 (1), 3–49.

Munch, K., Mailund, T., Dutheil, J.Y., Schierup, M.H., 2014. A fine-scale recombination map of the human-chimpanzee ancestor reveals faster change in humans than in chimpanzees and a strong impact of GC-biased gene conversion. Genome Res. 24 (3), 467–474, doi:10.1101/gr.158469.113.

Nelder, J.A., Mead, R., 1965. A simplex method for function minimization. Comput. J. 7 (4), 308–313, http://dx.doi.org/10.1093/comjnl/7.4.308 http://comjnl.oxfordjournals.org/content/7/4/308.abstract

Prüfer, K., Munch, K., Hellmann, I., Akagi, K., Miller, J.R., Walenz, B., Koren, S., Sutton, G., Kodira, C., Winer, R., Knight, J.R., Mullikin, J.C., Meader, S.J., Ponting, C.P., Lunter, G., Higashino, S., Hobolth, A., Dutheil, J., Karakoç, E., Alkan, C., Sajjadian, S., Catacchio, C.R., Ventura, M., Marquès-Bonet, T., Eichler, E.E., André, C., Atencia, R., Mugisha, L., Junhold, J., Patterson, N., Siebauer, M., Good, J.M., Fischer, A., Ptak, S.E., Lachmann, M., Symer, D.E., Mailund, T., Schierup, M.H., Andrés, A.M., Kelso, J., Pääbo, S., 2012. The bonobo genome compared with the chimpanzee and human genomes. Nature 486 (7404), 527–531.

Prado-Martinez, J., Sudmant, P.H., Kidd, H., Li, Jeffrey Mand, Kelley, J.L., Lorente-Galdos, B., Veeramah, K.R., Woerner, A.E., O'Connor, T.D., Santpere, G., Cagan, A., Theunert, C., Casals, F., Laayouni, H., Munch, K., Hobolth, A., Halager, A.E., Malig, M., Hernandez-Rodriguez, J., Hernando-Herraez, I., Prüfer, K., Pybus, M., Johnstone, L., Lachmann, M., Alkan, C., Twigg, D., Petit, N., Baker, C., Hormozdiari, F., Fernández-Callejo, M., Dabad, M., Wilson, M.L., Stevison, L., Camprubí, C., Carvalho, T., Ruiz-Herrera, A., Vives, L., Melé, M., Abello, T., Kondova, I., Bontrop, R.E., Pusey, A., Lankester, F., Kiyang, J.A., Bergl, R.A., Lonsdorf, E., Myers, S., Ventura, M., Gagneux, P., Comas, D., Siegismund, H., Blanc, J., Agueda-Calpena, L., Gut, M., Fulton, L., Tishkoff, S.A., Mullikin, J.C., Wilson, R.K., Gut, I.G., Gonder, M.K., Ryder, O.A., Hahn, B.H., Navarro, A., Akey, J.M., Bertranpetit, J., Reich, D., Mailund, T., Schierup, M.H., Hvilsom, C., Andrés, A.M., Wall, J.D., Bustamante, C.D., Hammer, M.F., Eichler, E.E., Marquès-Bonet, T., 2013. Great ape genetic diversity and population history. Nature 499 (7459), 471–475.

Rabiner, L., 1989. A tutorial on hidden markov models and selected applications in speech recognition. Proc. IEEE, 257–286, http://dx.doi.org/10.1109/5.18626.

Sand, A., Kristiansen, M., Pedersen, C.N., Mailund, T., 2013. Ziphmmlib: a highly optimised hmm library exploiting repetitions in the input to speed up the forward algorithm. BMC Bioinform. 14, 339, http://dx.doi.org/10.1186/1471-2105-14-339.

Scally, A., Dutheil, J.Y., Hillier, L.W., Jordan, G.E., Goodhead, I., Herrero, J., Hobolth, A., Lappalainen, T., Mailund, T., Marques-Bonet, T., McCarthy, S., Montgomery, S.H., Schwalie, P.C., Tang, Y.A., Ward, M.C., Xue, Y., Yngvadóttir, B., Alkan, C., Andersen, L.N., Ayub, Q., Ball, E.V., Beal, K., Bradley, B.J., Chen, Y., Clee, C.M., Fitzgerald,

S., Graves, T.A., Gu, Y., Heath, P., Heger, A., Karakoc, E., Kolb-Kokocinski, A., Laird, G.K., Lunter, G., Meader, S., Mort, M., Mullikin, J.C., Munch, K., O'Connor, T.D., Phillips, A.D., Prado-Martinez, J., Rogers, A.S., Sajjadian, S., Schmidt, D., Shaw, K., Simpson, J.T., Stenson, P.D., Turner, D.J., Vigilant, L., Vilella, A.J., Whitener, W., Zhu, B., Cooper, D.N., de Jong, P., Dermitzakis, E.T., Eichler, E.E., Flicek, P., Goldman, N., Mundy, N.I., Ning, Z., Odom, D.T., Ponting, C.P., Quail, M.A., Ryder, O.A., Searle, S.M., Warren, W.C., Wilson, R.K., Schierup, M.H., Rogers, J., Tyler-Smith, C., Durbin, R., 2012. Insights into hominid evolution from the gorilla genome sequence. Nature 483 (7388), 169–175.

Schiffels, S., Durbin, R., 2014. Inferring human population size and separation history from multiple genome sequences. Nat. Genet. 46 (8), 919–925.

Sheehan, S., Harris, K., Song, Y.S., 2013. Estimating variable effective population sizes from multiple genomes: a sequentially Markov conditional sampling distribution approach. Genetics 194 (3), 647–662.

Shi, Y., Eberhart, R., 1998. A modified particle swarm optimizer. In: IEEE World Congress on Computational Intelligence. The 1998 IEEE International Conference on Evolutionary Computation Proceedings, pp. 69–73, http://dx.doi.org/10.1109/ICEC.1998.699146.

Simonsen, K., Churchill, G., 1997. A Markov chain model of coalescence with recombination. Theor. Popul. Biol. 52 (1), 43–59.

Slatkin, M., Pollack, J.L., 2006. The concordance of gene trees and species trees at two linked loci. Genetics 172 (3), 1979–1984.

Steinrücken, M., Paul, J.S., Song, Y.S., 2013. A sequentially Markov conditional sampling distribution for structured populations with migration and recombination. Theor. Popul. Biol. 87, 51–61.

Syswerda, G., 1989. Uniform crossover in genetic algorithms. In: Schaffer, J.D. (Ed.), Proceedings of the Third International Conference on Genetic Algorithms. Morgan Kaufmann, pp. 2–9.